

Package ‘serad’

July 23, 2026

Title Standardized Economic Reporting and Automated Dynamic Writing /
Synthèse d'Écrits Avec des Règles Automatisées et Dynamiques

Version 0.2.3

Description Provides tools for generating dynamic and standardized economic narratives in R Markdown documents. The package is primarily designed for French-language statistical and economic publications. It includes functions to describe changes in levels, percentages, trends, accelerations and short-term economic developments using consistent linguistic rules. The package supports automated reporting workflows and reproducible economic writing.
Fournit des outils permettant de générer des textes économiques dynamiques et standardisés dans des documents R Markdown. Le package est principalement conçu pour les publications statistiques et économiques en français. Il propose des fonctions permettant de décrire les évolutions de niveaux, de pourcentages, de tendances, d'accélération et les évolutions conjoncturelles à l'aide de règles linguistiques homogènes. Le package facilite l'automatisation de la rédaction et la reproductibilité des publications économiques.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports utils, tibble

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Alexandre Cazenave-Lacroutz [aut] (ORCID:
<<https://orcid.org/0000-0003-3488-6092>>),
Jules Lejas [cre],
Direction de l'animation de la recherche, des études et des
statistiques (Dares) [cph]

Maintainer Jules Lejas <jules.lejas@gmail.com>

Repository CRAN

Date/Publication 2026-07-23 13:50:02 UTC

Contents

a	3
alahausse	4
arrondi_tot	4
audessus	5
comparaison	6
comparaison_taux	7
contributions	8
davantage	9
depasse	10
format_delta	10
format_g	11
format_niv	12
format_pts	13
g	14
gETa_nom	15
gETa_nom_taux	16
gETa_verbe	17
gETa_verbe_taux	18
get_serad_language	20
g_nom	20
g_nom_simple	21
g_nom_taux	22
g_verbe	23
g_verbe_taux	25
init_serad	26
nextMois	27
nextTrim	28
plushautniveau	29
prevMois	30
prevTrim	31
quelMois	32
quelTrim	33
s	35
set_serad_language	36
whichMois	36

Index

37

a	<i>Calcul d'une accélération</i>
---	----------------------------------

Description

Calcule l'accélération entre trois niveaux successifs en comparant les taux de variation consécutifs.

Usage

`a(x1, x2, x3)`

Arguments

x1	Niveau le plus récent.
x2	Niveau précédent.
x3	Niveau le plus ancien.

Details

L'accélération correspond à la variation du taux entre (x1, x2) et (x2, x3).

Value

Un nombre numérique correspondant à l'accélération en pourcentage.

See Also

[g](#), [format_g](#)

Examples

```
a(4, 2, 1) # 0
a(6, 2, 1) # 100
a(2, 1, 1) # valeur très élevée si taux précédent proche de zéro
```

alahausse	<i>Évolution à la hausse, à la baisse ou inchangée</i>
-----------	--

Description

Indique si x1 évolue à la hausse, à la baisse ou reste inchangé par rapport à x2.

Usage

```
alahausse(x1, x2, seuil = 0.1, lang = get_serad_language())
```

Arguments

x1	Niveau le plus récent.
x2	Niveau le plus ancien.
seuil	Seuil d'égalité en valeur absolue.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères.

See Also

[comparaison](#)

arrondi_tot	<i>Arrondi arithmétique</i>
-------------	-----------------------------

Description

Arrondit un nombre selon la règle arithmétique (0.5 vers le haut), contrairement à round() qui utilise l'arrondi bancaire.

Usage

```
arrondi_tot(x, digits = 1)
```

Arguments

x	Nombre à arrondir.
digits	Entier indiquant le nombre de décimales. Positif pour les décimales, négatif pour les dizaines, centaines, etc. Par défaut : 1.

Value

Un nombre numérique arrondi.

See Also

[format_niv](#)

Examples

```
arrondi_tot(1877.85, digits = 0) # 1878
arrondi_tot(1877.85, digits = 1) # 1877.9
arrondi_tot(1877.85, digits = -1) # 1880
```

audessus	<i>Position relative entre deux niveaux</i>
----------	---

Description

Indique si x1 est au-dessus ou en dessous de x2.

Usage

```
audessus(x1, x2, lang = get_serad_language())
```

Arguments

x1	Niveau le plus récent.
x2	Niveau le plus ancien.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères.

See Also

[comparaison](#)

comparaison

Comparaison qualitative entre deux niveaux

Description

Compare deux niveaux successifs et retourne une formulation selon l'évolution observée.

Usage

```
comparaison(  
  x1,  
  x2,  
  hausse_default,  
  egalite_default,  
  baisse_default,  
  seuil = 0.1,  
  alt = 0,  
  hausse_alt = hausse_default,  
  egalite_alt = egalite_default,  
  baisse_alt = baisse_default  
)
```

Arguments

x1	Niveau le plus récent.
x2	Niveau le plus ancien.
hausse_default	Formulation en cas de hausse.
egalite_default	Formulation en cas de stabilité.
baisse_default	Formulation en cas de baisse.
seuil	Seuil d'égalité en valeur absolue. Par défaut : 0.1.
alt	Indicateur logique permettant d'utiliser une formulation alternative.
hausse_alt	Formulation alternative en cas de hausse.
egalite_alt	Formulation alternative en cas de stabilité.
baisse_alt	Formulation alternative en cas de baisse.

Details

La comparaison repose sur le taux de variation calculé via [g](#). Des cas particuliers sont traités lorsque x2 est nul ou négatif.

Value

Une chaîne de caractères correspondant à la formulation retenue.

See Also

[comparaison_taux, g](#)

Examples

```
comparaison(1.04, 1, "augmente", "reste stable", "diminue")
comparaison(0.9991, 1, "augmente", "reste stable", "diminue")
comparaison(1, 1, "augmente", "reste égal", "diminue", seuil = 0)
```

comparaison_taux	<i>Comparaison d'une variation à un seuil</i>
------------------	---

Description

Comparaison d'une variation à un seuil

Usage

```
comparaison_taux(  
  g,  
  hausse_default,  
  egalite_default,  
  baisse_default,  
  seuil = 0.1,  
  alt = 0,  
  hausse_alt = hausse_default,  
  egalite_alt = egalite_default,  
  baisse_alt = baisse_default  
)
```

Arguments

<code>g</code>	Variation exprimée en pourcentage (5 signifie 5 %, 0.1 signifie 0.1 %).
<code>hausse_default</code>	Mot si hausse (forme par défaut).
<code>egalite_default</code>	Mot si égalité (forme par défaut).
<code>baisse_default</code>	Mot si baisse (forme par défaut).
<code>seuil</code>	Limite pour l'égalité (0.1 par défaut).
<code>alt</code>	Paramètre supplémentaire égal à 0 ou 1 (par exemple pour distinguer singulier/pluriel).
<code>hausse_alt</code>	Formulation alternative si <code>alt = 1</code> .
<code>egalite_alt</code>	Formulation alternative si <code>alt = 1</code> .
<code>baisse_alt</code>	Formulation alternative si <code>alt = 1</code> .

Details

Fonction interne utilisée par [comparaison](#).

Value

Une chaîne de caractères correspondant à la modalité choisie.

See Also

[comparaison](#)

Examples

```
comparaison_taux(5, "augmente", "reste stable", "diminue")
comparaison_taux(0.05, "augmente", "reste stable", "diminue")
comparaison_taux(0, "as", "bs", "cs", seuil = 0, alt = 1,
                 hausse_alt = "a", egalite_alt = "b", baisse_alt = "c")
```

contributions

Documente les contributions à une évolution

Description

Un dataframe en entrée.

Si non indiqué, la date est considérée comme étant la première colonne. Pas besoin de la formater.

Si non indiqué, on suppose les valeurs les plus anciennes en haut (temps croissant).

Toutes les autres colonnes sont considérées comme les composantes dont les contributions doivent être analysées. Elles doivent être numériques.

Usage

```
contributions(
  df,
  temps = "croissant",
  vart,
  Tglissement = 1,
  seuilpc = 20,
  lang = get_serad_language()
)
```

Arguments

df	Le dataframe en entrée
temps	"croissant" par défaut si du passé vers le présent. "décroissant" sinon.
vart	La variable de temps du dataframe. La première valeur par défaut.
Tglissement	Sur combien de périodes faut-il remonter pour calculer l'évolution à 1 par défaut.
seuilpc	Un seuil en dessous duquel la contribution (en valeur absolue) n'est pas prise en compte. 5 % par défaut.
lang	Langue de sortie : "fr" ou "en".

Details

Fonction écrite sur demande d'une utilisatrice.

Si aucune contribution n'est supérieure à seuilpc en valeur absolue, contributions() retourne a minima la plus grande contribution de même sens que la contribution totale.

Value

Un descriptif ordonné des différentes glissements.

Examples

```
col0 = c("Y1T1", "Y1T2", "Y1trim3", "Y1T4", "Y2T1", "Y2-T2")
col1 = c( 12,      6,      2,      86,      19,      10)
col2 = c(  4,      8,      7,      34,      87,      14)
col3 = c( 10,     20,      3,      66,      90,      54)
col4 = c( 29,     12,      4,      16,      40,      94)
col5 = c( 58,     76,      1,       3,      34,      19)
df1 = data.frame(col0,col1,col2,col3,col4,col5)

contributions(df1)
```

davantage

Davantage ou moins

Description

Indique si x1 est davantage ou moins élevé que x2.

Usage

```
davantage(x1, x2, lang = get_serad_language())
```

Arguments

x1	Niveau le plus récent.
x2	Niveau le plus ancien.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères.

See Also

[comparaison](#)

depasse

Verbe pour exprimer le sens de l'évolution

Description

Indique si x1 excède, est au niveau de ou est en dessous de x2, en tenant compte du nombre grammatical.

Usage

```
depasse(x1, x2, sing = FALSE, lang = get_serad_language())
```

Arguments

x1	Le niveau le plus récent.
x2	Le niveau le plus ancien.
sing	Indicateur logique : FALSE si le sujet est pluriel, TRUE s'il est singulier (par défaut : pluriel).
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères.

See Also

[comparaison](#)

format_delta*Formatage des variations en niveau*

Description

Formate une différence numérique selon les règles d'arrondi et d'affichage utilisées dans le package.

Usage

```
format_delta(  
  y,  
  signe = TRUE,  
  detail = getOption("serad")$arrondi_niv,  
  lang = get_serad_language()  
)
```

Arguments

y	La différence à formater.
signe	Indicateur logique : TRUE pour afficher le signe, FALSE sinon (par défaut : TRUE).
detail	Précision d'arrondi. Par défaut, on utilise <code>getOption("serad")\$arrondi_niv</code> .
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant à la variation formatée.

See Also

[arrondi_tot](#), [format_niv](#)

Examples

```
format_delta(365484)           # "+365 500"
format_delta(365484, FALSE)    # "365 500"
format_delta(-365484)          # "-365 500"
format_delta(-365484, FALSE)   # "365 500"
format_delta(365484, lang = "en") # "+365,500"
```

format_g

Formatage des variations en pourcentage

Description

Formate une variation exprimée en pourcentage selon les règles d'arrondi et d'affichage du package.

Usage

```
format_g(
  y,
  signe = TRUE,
  detail = getOption("serad")$arrondi_pourcent,
  lang = get_serad_language()
)
```

Arguments

y	La variation à formater.
signe	Indicateur logique : TRUE pour afficher le signe, FALSE pour le retirer (par défaut : TRUE).
detail	Nombre de chiffres après la virgule. Par défaut, on utilise <code>getOption("serad")\$arrondi_pourcent</code> .
lang	Langue de sortie : "fr" ou "en".

Details

Le symbole "moins" peut être personnalisé via `getOption("serad")$moins`.

Value

Une chaîne de caractères correspondant à la variation formatée (ex. "+5,4%", "-5,4%", "+5.4%").

See Also

[g, arrondi_tot](#)

Examples

```
format_g(5.3654, signe = FALSE)      # "5,4 %"
format_g(5.3654)                     # "+5,4 %"
format_g(-5.3654, FALSE)             # "5,4 %"
format_g(-5.3654)                    # "-5,4 %"
format_g(-5.3654, detail = 2)        # "-5,37 %"
format_g(0.35)                       # "+0,4 %"
format_g(5.3654, lang = "en")        # "+5.4%"
```

format_niv

Formatage des niveaux en nombres

Description

Formate un niveau numérique selon les règles d'arrondi définies dans le package.

Usage

```
format_niv(
  y,
  detail = getOption("serad")$arrondi_niv,
  lang = get_serad_language()
)
```

Arguments

y	Le niveau à formater.
detail	Précision d'arrondi. Par défaut, on utilise <code>getOption("serad")\$arrondi_niv</code> .
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant au niveau formaté.

See Also[arrondi_tot](#)**Examples**

```
format_niv(365484)          # "365 500"  
format_niv(365484, lang = "en") # "365,500"
```

format_pts*Formatage des variations en points*

Description

Formate une variation exprimée en points selon les règles d'arrondi et d'affichage du package.

Usage

```
format_pts(  
  y,  
  signe = TRUE,  
  detail = getOption("serad")$arrondi_pourcent,  
  abrev = FALSE,  
  lang = get_serad_language()  
)
```

Arguments

y	La variation à formater.
signe	Indicateur logique : TRUE pour afficher le signe, FALSE pour le retirer (par défaut : TRUE).
detail	Nombre de chiffres après la virgule. Par défaut, on utilise getOption("serad")\$arrondi_pourcent.
abrev	Indicateur logique : TRUE pour utiliser l'abréviation ("pt"/"pts"), FALSE pour afficher "point(s)" (par défaut : FALSE).
lang	Langue de sortie : "fr" ou "en".

Details

Le symbole "moins" peut être personnalisé via getOption("serad")\$moins.

Value

Une chaîne de caractères correspondant à la variation formatée (ex. "+5,4 points", "+5.4 points").

See Also[format_g](#)

Examples

```
format_pts(5.3654, signe = FALSE) # "5,4 points"
format_pts(1.3654, signe = FALSE) # "1,4 point"
format_pts(5.3654, abrev = TRUE)  # "+5,4 pts"
format_pts(-5.3654, FALSE)        # "5,4 points"
format_pts(-5.3654)                # "-5,4 points"
format_pts(-5.3654, detail = 2)   # "-5,37 points"
format_pts(0.35)                  # "+0,4 points"
```

g

Calcul d'une variation relative

Description

Calcule la variation relative entre x1 et x2, exprimée en pourcentage.

Usage

```
g(x1, x2, eps = 1e-08)
```

Arguments

x1	Niveau le plus récent.
x2	Niveau le plus ancien.
eps	Valeur utilisée à la place de x2 lorsque x2 = 0, afin d'éviter une division par zéro. Par défaut : 1e-8.

Details

Si x2 = 0, la valeur epsilon définie par `getOption("serad")$eps` est utilisée afin d'éviter une division par zéro. Un message d'avertissement est émis dans ce cas.

Il est possible de modifier cette valeur :

```
serad0 <- getOption("serad") serad0$eps <- 0 options(serad = serad0)
```

Value

Une valeur numérique correspondant à la variation en pourcentage (par exemple, si $x1 = 2 * x2$, la fonction retourne 100).

See Also

[format_g](#)

Examples

```
g(2, 1) # 100
g(2, 0) # valeur très élevée et avertissement
```

gETa_nom*Évolution nominale tenant compte de l'accélération*

Description

Décrit l'évolution sous forme nominale à partir de trois niveaux, en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_nom(x1, x2, x3, titre = FALSE, alea = 0, lang = get_serad_language())
```

Arguments

x1	Niveau le plus récent.
x2	Niveau précédent.
x3	Niveau le plus ancien.
titre	Indicateur logique : TRUE pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
alea	Paramètre numérique compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si alea = 0, la formulation est déterministe. Si alea = 1, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord deux évolutions successives : `g1 <- serad : g(x1, x2)` et `g2 <- serad : g(x2, x3)`.

Ces deux variations sont ensuite transmises à [gETa_nom_taux](#), qui calcule leur accélération à l'aide de [a](#) et détermine la formulation nominale appropriée à partir de la table `getOption("serad")$evo_accel`.

Une formulation alternative peut être utilisée via la table `getOption("serad")$evo_accel_alt`.

Le choix entre la formulation principale et la variante dépend du paramètre `alea`.

Value

Une chaîne de caractères correspondant à la formulation nominale retenue.

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables `getOption("serad")$evo_accel` et `getOption("serad")$evo_accel_alt`.

Pour modifier les seuils, les conditions ou les libellés, voir [init_serad](#).

See Also

[gETa_nom_taux](#), [g](#), [a](#), [init_serad](#)

Examples

```
gETa_nom(1.00049, 1, 0.9996)
gETa_nom(1.1, 1, 0.99)
gETa_nom(1.003, 1, 0.99)
gETa_nom(0.96, 1, 1.01)
gETa_nom(0.8, 1, 0.99)
gETa_nom(1.1, 1, 0.99, alea = 0.5)
```

gETa_nom_taux	<i>Évolution nominale tenant compte de l'accélération</i>
---------------	---

Description

Décrit l'évolution sous forme nominale en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_nom_taux(g1, g2, titre = FALSE, alea = 0, lang = get_serad_language())
```

Arguments

g1	Dernière évolution, exprimée en pourcentage.
g2	Évolution précédente, exprimée en pourcentage.
titre	Indicateur logique : TRUE pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
alea	Paramètre numérique compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si alea = 0, la formulation est déterministe. Si alea = 1, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord l'accélération entre g1 et g2 à l'aide de [a](#).

Elle parcourt ensuite la table `getOption("serad")$evo_accel` ligne par ligne. Chaque ligne contient un ensemble de conditions sur g1, g2 et l'accélération. La première ligne dont toutes les conditions sont vérifiées détermine la formulation retenue.

Une formulation alternative peut être utilisée via la table `getOption("serad")$evo_accel_alt`, qui contient une variante pour chaque ligne de `evo_accel`. Le choix entre la formulation principale et la variante dépend du paramètre `alea`.

Si `titre = TRUE`, l'article initial est supprimé et la première lettre restante est mise en majuscule.

Value

Une chaîne de caractères correspondant à la formulation nominale retenue (par exemple : "une accélération", "une stabilisation").

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables `getOption("serad")$evo_accel` et `getOption("serad")$evo_accel_alt`.

Pour modifier les seuils, les conditions ou les libellés, voir [init_serad](#).

See Also

[a](#), [gETa_verbe_taux](#), [init_serad](#)

Examples

```
gETa_nom_taux(0.049, 0.049)
gETa_nom_taux(10, 1)
gETa_nom_taux(-4, 1, titre = TRUE)
gETa_nom_taux(-21, 1)
gETa_nom_taux(10, 1, alea = 0.5)
```

gETa_verbe

Évolution verbale tenant compte de l'accélération

Description

Décrit l'évolution sous forme verbale à partir de trois niveaux, en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_verbe(x1, x2, x3, sing = TRUE, alea = 0, lang = get_serad_language())
```

Arguments

x1	Niveau le plus récent.
x2	Niveau précédent.
x3	Niveau le plus ancien.
sing	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.
alea	Paramètre numérique compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si alea = 0, la formulation est déterministe. Si alea = 1, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord deux évolutions successives : `g1 <- serad::g(x1, x2)` et `g2 <- serad::g(x2, x3)`.

Ces deux variations sont ensuite transmises à `gETa_verbe_taux`, qui calcule leur accélération à l'aide de `g` et détermine la formulation verbale appropriée à partir de la table `getOption("serad")$evo_accel`.

Une formulation alternative peut être utilisée via la table `getOption("serad")$evo_accel_alt`. Le choix entre la formulation principale et la variante dépend du paramètre `alea`.

Value

Une chaîne de caractères correspondant à la formulation verbale retenue.

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables `getOption("serad")$evo_accel` et `getOption("serad")$evo_accel_alt`.

Pour modifier les seuils, les conditions ou les libellés, voir `init_serad`.

See Also

`gETa_verbe_taux`, `g`, `init_serad`

Examples

```
gETa_verbe(1.00049, 1, 0.9996)
gETa_verbe(1.1, 1, 0.99)
gETa_verbe(1.003, 1, 0.99, sing = FALSE)
gETa_verbe(0.96, 1, 1.01)
gETa_verbe(1.1, 1, 0.99, alea = 0.5)
```

`gETa_verbe_taux`

Évolution verbale tenant compte de l'accélération

Description

Décrit l'évolution sous forme verbale en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_verbe_taux(g1, g2, sing = TRUE, alea = 0, lang = get_serad_language())
```

Arguments

g1	Dernière évolution, exprimée en pourcentage.
g2	Évolution précédente, exprimée en pourcentage.
sing	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.
alea	Nombre réel compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si alea = 0, la formulation est déterministe. Si alea = 1, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord l'accélération entre g1 et g2 à l'aide de [a](#).

Elle parcourt ensuite la table `getOption("serad")$evo_accel` ligne par ligne. Chaque ligne contient un ensemble de conditions sur g1, g2 et l'accélération. La première ligne pour laquelle toutes les conditions sont vérifiées détermine la formulation retenue.

Une formulation alternative peut être utilisée via la table `getOption("serad")$evo_accel_alt`. Le choix entre la formulation principale et la variante dépend du paramètre alea.

Si sing = TRUE, la fonction renvoie la colonne `verbe_sing`. Sinon, elle renvoie `verbe_plur`.

Value

Une chaîne de caractères correspondant à la formulation verbale retenue (par exemple : "accélère", "se stabilise").

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables `getOption("serad")$evo_accel` et `getOption("serad")$evo_accel_alt`.

Pour modifier les seuils, les conditions ou les libellés, voir [init_serad](#).

See Also

[a](#), [gETa_nom_taux](#), [init_serad](#)

Examples

```
gETa_verbe_taux(0.049, 0.049)
gETa_verbe_taux(10, 1)
gETa_verbe_taux(4, 1, FALSE)
gETa_verbe_taux(-4, 1)
gETa_verbe_taux(-21, 1)
gETa_verbe_taux(10, 1, alea = 0.5)
```

get_serad_language	Récupère la langue de serad
--------------------	-----------------------------

Description

Retourne la langue actuellement utilisée par {serad}.

Usage

```
get_serad_language()
```

Value

Une chaîne de caractères indiquant la langue courante ("fr" ou "en").

g_nom	Évolution nominale non suivie d'une valeur
-------	--

Description

Décrit une évolution sous forme nominale à partir de deux niveaux, sans ajouter la valeur de variation.

Usage

```
g_nom(  
  x1,  
  x2,  
  evolution = c("pourcents", "points"),  
  titre = FALSE,  
  lang = get_serad_language()  
)
```

Arguments

x1	Le niveau le plus récent.
x2	Le niveau le plus ancien.
evolution	Type d'évolution : "pourcents" (variation relative, par défaut) ou "points".
titre	Indicateur logique : TRUE pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord une évolution à partir de x_1 et x_2 :

- si `evolution = "pourcents"`, elle utilise `g` ;
- si `evolution = "points"`, elle calcule $x_1 - x_2$.

La valeur obtenue est ensuite transmise à `g_nom_taux`, qui détermine la formulation à partir des conditions définies dans la table `getOption("serad")$evo_simple`.

Value

Une chaîne de caractères correspondant à la formulation nominale retenue (par exemple : "une forte hausse").

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Cette table doit notamment contenir une colonne `condition` et une colonne `nom`. Les conditions doivent être disjointes, c'est-à-dire qu'une seule condition doit être vraie pour une valeur donnée.

Pour modifier les conditions ou les libellés, voir `init_serad`.

See Also

`g_nom_taux`, `g`, `init_serad`

Examples

```
g_nom(1.04, 1)
g_nom(1.01, 1)
g_nom(1.004, 1)
g_nom(1.001, 1)
g_nom(1, 1)
g_nom(0.997, 1)
g_nom(0.95, 1)
g_nom(0.95, 1, evolution = "points")
```

`g_nom_simple`*Évolution nominale simple*

Description

Décrit l'évolution entre x_1 et x_2 sous forme nominale suivie du pourcentage correspondant.

Usage

```
g_nom_simple(x1, x2, lang = get_serad_language())
```

Arguments

x1	Le niveau le plus récent.
x2	Le niveau le plus ancien.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères.

g_nom_taux	<i>Évolution nominale d'un taux</i>
------------	-------------------------------------

Description

Décrit une évolution exprimée en pourcentage sous forme nominale (par exemple : "une forte hausse").

Usage

```
g_nom_taux(g, titre = FALSE, lang = get_serad_language())
```

Arguments

g	L'évolution en pourcentage.
titre	Indicateur logique : TRUE pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction sélectionne, dans la table `getOption("serad")$evo_simple`, la ligne dont la condition est vérifiée par la valeur de `g`.

La table `evo_simple` doit contenir une colonne `condition`, composée de chaînes de caractères évaluable par R, par exemple : `"g >= -0.10 & g <= 0.10"`.

Les conditions doivent être disjointes : pour une valeur donnée de `g`, une seule condition doit être vraie.

Une catégorie est considérée comme une stabilité si sa condition est également vérifiée pour `g = 0`.

La fonction renvoie ensuite la colonne `nom` correspondante. Si `titre = TRUE`, l'article initial est supprimé et la première lettre restante est mise en majuscule.

Value

Une chaîne de caractères décrivant l'évolution (par exemple : "une forte hausse", "une stabilité").

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Pour modifier les conditions ou les libellés, voir [init_serad](#).

See Also

[g_verbe_taux](#), [init_serad](#)

Examples

```
g_nom_taux(4)
g_nom_taux(1)
g_nom_taux(0.4)
g_nom_taux(0.1)
g_nom_taux(0)
g_nom_taux(-0.1)
g_nom_taux(-0.3)
g_nom_taux(-1)
g_nom_taux(-4)
g_nom_taux(-5)
```

g_verbe	<i>Évolution verbale ne tenant pas compte de l'accélération et suivie de la valeur</i>
---------	--

Description

Décrit une évolution sous forme verbale à partir de deux niveaux, sans tenir compte d'une éventuelle accélération, et en ajoutant la valeur de variation.

Usage

```
g_verbe(
  x1,
  x2,
  sing = TRUE,
  evolution = c("pourcents", "points"),
  stable_sans_valeur = TRUE,
  lang = get_serad_language()
)
```

Arguments

x1	Le niveau le plus récent.
x2	Le niveau le plus ancien.
sing	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.

evolution	Type d'évolution : "pourcents" (variation relative, par défaut) ou "points".
stable_sans_valeur	Indicateur logique. TRUE (par défaut) : n'affiche pas la valeur en cas de stabilité. FALSE : affiche la valeur après la formulation de stabilité.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord une évolution à partir de x1 et x2 :

- si evolution = "pourcents", elle utilise [g](#) ;
- si evolution = "points", elle calcule $x1 - x2$.

La valeur obtenue est ensuite transmise à [g_verbe_taux](#), qui détermine la formulation à partir des conditions définies dans la table `getOption("serad")$evo_simple`.

Cette table doit contenir notamment les colonnes condition, verbe_sing, verbe_plur et nom.

Les conditions doivent être disjointes : pour une valeur donnée, une seule condition doit être vraie.

Value

Une chaîne de caractères correspondant à la formulation verbale retenue, par exemple : "augmente de 10,0 %".

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Pour modifier les conditions ou les libellés, voir [init_serad](#).

See Also

[g_verbe_taux](#), [g](#), [init_serad](#)

Examples

```
g_verbe(1.1, 1)
g_verbe(1.04, 1)
g_verbe(1.01, 1, sing = FALSE)
g_verbe(1.003, 1)
g_verbe(0.999, 1)
g_verbe(0.999, 1, stable_sans_valeur = FALSE)
g_verbe(0.96, 1)
g_verbe(0.79, 1)
```

g_verbe_taux	<i>Évolution verbale d'un taux</i>
--------------	------------------------------------

Description

Décrit une évolution sous forme verbale, sans tenir compte d'une éventuelle accélération, et suivie de la valeur formatée.

Usage

```
g_verbe_taux(  
  g,  
  sing = TRUE,  
  evolution = c("pourcents", "points"),  
  stable_sans_valeur = TRUE,  
  lang = get_serad_language()  
)
```

Arguments

g	L'évolution.
sing	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.
evolution	Type d'évolution : "pourcents" (variation relative) ou "points".
stable_sans_valeur	Indicateur logique : TRUE (par défaut) pour ne rien ajouter après une stabilité. Si FALSE, ajoute la valeur après la formulation de stabilité.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction sélectionne, dans la table `getOption("serad")$evo_simple`, la ligne dont la condition est vérifiée par la valeur de `g`.

La table `evo_simple` doit contenir une colonne `condition`, composée de chaînes de caractères évaluable par R, par exemple : `"g >= -0.10 & g <= 0.10"`.

Les conditions doivent être disjointes : pour une valeur donnée de `g`, une seule condition doit être vraie.

Si `sing = TRUE`, la fonction renvoie la colonne `verbe_sing`. Sinon, elle renvoie `verbe_plur`.

Value

Une chaîne de caractères décrivant l'évolution.

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Pour modifier les conditions ou les libellés, voir [init_serad](#).

See Also

[g_verbe](#), [format_g](#), [format_pts](#), [init_serad](#)

Examples

```
g_verbe_taux(10)
g_verbe_taux(0.1)
g_verbe_taux(-0.1)
g_verbe_taux(-0.1, stable_sans_valeur = FALSE)
```

init_serad

Initialisation des règles de rédaction de {serad}

Description

Les fonctions `init_serad_fr()` et `init_serad_en()` définissent l'ensemble des règles utilisées par {serad} pour produire des textes de conjoncture.

Elles initialisent notamment :

- les règles d'arrondi ;
- le symbole de signe négatif ;
- les seuils d'évolution ;
- les tables de correspondance entre évolutions et formulations ;
- les règles de gestion de l'accélération.

Usage

```
init_serad_en()
```

```
init_serad_fr()
```

Details

Les principales structures utilisées sont :

- `evo_simple` : utilisée pour les évolutions simples ([g_nom_taux](#), [g_verbe_taux](#)) ;
- `evo_accel` : utilisée pour les évolutions tenant compte de l'accélération ([gETa_nom_taux](#), [gETa_verbe_taux](#)) ;
- `evo_accel_alt` : variantes utilisées lorsque l'argument `alea` est activé.

Value

Pas de valeur de retour, appelée pour ses effets de bord.

Personnalisation

La méthode recommandée pour personnaliser les règles de rédaction consiste à copier le contenu de `init_serad_fr()` ou `init_serad_en()` dans un script utilisateur, puis à modifier directement les tables et les seuils.

Exemple :

```
# Copier le contenu de init_serad_fr()

serad0 <- list()

serad0$evo_simple <- tibble::tribble(
  ~seuil, ~verbe_sing, ~verbe_plur, ~nom,
  1, "augmente", "augmentent", "une hausse",
  0, "est stable", "sont stables", "une stabilité",
  -Inf, "diminue", "diminuent", "une baisse"
)

options(serad = serad0)
```

Cette approche permet :

- une personnalisation complète des formulations ;
- une adaptation aux conventions métier ;
- une reproductibilité via un script versionné.

See Also

[g_nom_taux](#), [g_verbe_taux](#), [gETa_nom_taux](#), [gETa_verbe_taux](#)

nextMois

Fournit dynamiquement le mois suivant

Description

Retourne la formulation du mois situé à *k* périodes après le mois indiqué.

Usage

```
nextMois(
  mois,
  annee,
  type = c("Annee", "autres"),
  majuscule = FALSE,
  k = 1,
  lang = get_serad_language()
)
```

Arguments

mois	Mois sous forme numérique : 1 à 12.
annee	Année sur 4 chiffres.
type	"Annee" (par défaut) ou "autres".
majuscule	Indicateur logique : TRUE pour une majuscule initiale, FALSE sinon.
k	Décalage en nombre de mois. Par défaut : 1 (mois suivant).
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant au mois suivant.

See Also

[quelMois](#), [prevMois](#), [nextTrim](#)

Examples

```
nextMois(3, 2023)
nextMois(12, 2023)
nextMois(12, 2023, k = 2)
```

nextTrim

Fournit dynamiquement le trimestre suivant

Description

Retourne la formulation du trimestre situé à k périodes après le trimestre indiqué.

Usage

```
nextTrim(
  trim,
  annee,
  type = c("lettres", "chiffres"),
  majuscule = FALSE,
  exposant = TRUE,
  k = 1,
  lang = get_serad_language()
)
```

Arguments

trim	Trimestre sous forme numérique : 1, 2, 3 ou 4.
annee	Année sur 4 chiffres.
type	"lettres" (par défaut) ou "chiffres".
majuscule	Indicateur logique : TRUE pour une majuscule initiale, FALSE sinon.
exposant	Indicateur logique : TRUE pour afficher les exposants Markdown (ex. "1 ^{er} "), FALSE pour afficher "1er".
k	Décalage en nombre de trimestres. Par défaut : 1 (trimestre suivant).
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant au trimestre suivant.

See Also

[quelTrim](#), [prevTrim](#)

Examples

```
nextTrim(3, 2023)
nextTrim(4, 2023)
```

plushautniveau	<i>Indique si le niveau est le plus haut ou le plus bas de la série</i>
----------------	---

Description

Détermine si la dernière valeur d'une série correspond au plus haut ou au plus bas niveau observé depuis un certain nombre de périodes.

Usage

```
plushautniveau(
  df,
  temps = "croissant",
  voc_haut,
  voc_bas,
  vart,
  vary,
  nbperiode = 3,
  lang = get_serad_language()
)
```

Arguments

df	Data frame en entrée.
temps	Ordre temporel : "croissant" (par défaut) si les valeurs vont du passé vers le présent, "décroissant" sinon.
voc_haut	Formulation utilisée pour un plus haut niveau. Si omis, dépend de la langue.
voc_bas	Formulation utilisée pour un plus bas niveau. Si omis, dépend de la langue.
var_t	Variable de temps du data frame. Par défaut : première colonne.
var_y	Variable numérique du data frame. Par défaut : seconde colonne.
nbperiode	Nombre minimal de périodes depuis lesquelles le niveau doit être un extremum pour retourner une phrase (par défaut : 3).
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères indiquant le plus haut ou le plus bas niveau depuis une date donnée.
Retourne une chaîne vide si rien de notable.

Exemples

```
col0 <- c("Y1T1", "Y1T2", "Y1trim3", "Y1T4", "Y2T1", "Y2-T2")
col1 <- c(12, 11, 7, 6, 9, 10)
col2 <- c(12, 11, 7, 6, 9, 14)
col3 <- c(12, 11, 3, 6, 9, 4)
col4 <- c(12, 11, 7, 6, 9, 4)
col5 <- c(12, 11, 7, 3, 9, 4)
df1 <- data.frame(col0, col1, col2, col3, col4, col5)

plushautniveau(df1)
plushautniveau(df1, nbperiode = 5)
plushautniveau(df1, vary = "col2")
plushautniveau(df1, vary = "col3")
plushautniveau(df1, vary = "col4")
```

prevMois

Fournit dynamiquement le mois précédent

Description

Retourne la formulation du mois situé à k périodes avant le mois indiqué.

Usage

```
prevMois(  
  mois,  
  annee,  
  type = c("Annee", "autres"),  
  majuscule = FALSE,  
  k = 1,  
  lang = get_serad_language()  
)
```

Arguments

mois	Mois sous forme numérique : 1 à 12.
annee	Année sur 4 chiffres.
type	"Annee" (par défaut) ou "autres".
majuscule	Indicateur logique : TRUE pour une majuscule initiale, FALSE sinon.
k	Décalage en nombre de mois. Par défaut : 1 (mois précédent).
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant au mois précédent.

See Also

[prevTrim](#), [nextMois](#), [quelMois](#)

Examples

```
prevMois(1, 2023)  
prevMois(12, 2023, k = 2)
```

prevTrim

Fournit dynamiquement le trimestre précédent

Description

Retourne la formulation du trimestre situé à k périodes avant le trimestre indiqué.

Usage

```
prevTrim(  
  trim,  
  annee,  
  type = c("lettres", "chiffres"),  
  majuscule = FALSE,  
  exposant = TRUE,  
  k = 1,  
  lang = get_serad_language()  
)
```

Arguments

trim	Trimestre sous forme numérique : 1, 2, 3 ou 4.
annee	Année sur 4 chiffres.
type	"lettres" (par défaut) ou "chiffres".
majuscule	Indicateur logique : TRUE pour une majuscule initiale, FALSE sinon.
exposant	Indicateur logique : TRUE pour afficher les exposants Markdown (ex. "1 ^{er} "), FALSE pour afficher "1er".
k	Décalage en nombre de trimestres. Par défaut : 1 (trimestre précédent).
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant au trimestre précédent.

See Also

[nextTrim](#), [quelTrim](#)

Examples

```
prevTrim(1, 2023)  
prevTrim(1, 2023, type = "chiffres", exposant = FALSE)
```

quelMois

Fournit dynamiquement le mois en toutes lettres

Description

Retourne une formulation littéraire du mois, avec ou sans année.

Usage

```
quelMois(  
  mois,  
  annee,  
  type = c("Annee", "autres"),  
  majuscule = FALSE,  
  lang = get_serad_language()  
)
```

Arguments

mois	Mois sous forme numérique : 1 à 12.
annee	Année sur 4 chiffres.
type	"Annee" (par défaut, ex. "janvier 2023") ou "autres" (ex. "janvier").
majuscule	Indicateur logique : TRUE pour une majuscule initiale (ex. "Janvier"), FALSE sinon.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant au mois, avec ou sans année.

See Also

[quelTrim](#), [nextMois](#), [prevMois](#)

Examples

```
quelMois(3, 2023)  
quelMois(3, 2023, majuscule = TRUE)  
quelMois(3, 2023, type = "autres")
```

quelTrim

Fournit dynamiquement le trimestre en toutes lettres ou en chiffres

Description

Retourne une formulation littéraire du trimestre pour une année donnée.

Usage

```
quelTrim(  
  trim,  
  annee,  
  type = c("lettres", "chiffres"),  
  majuscule = FALSE,  
  exposant = TRUE,  
  mois = NULL,  
  lang = get_serad_language()  
)
```

Arguments

trim	Trimestre sous forme numérique : 1, 2, 3 ou 4.
annee	Année sur 4 chiffres.
type	"lettres" (par défaut) ou "chiffres".
majuscule	Indicateur logique : TRUE pour une majuscule initiale (ex. "Premier"), FALSE sinon.
exposant	Indicateur logique : TRUE pour afficher les exposants Markdown (ex. "1 ^{er} "), FALSE pour afficher "1er".
mois	Si renseigné (entier entre 1 et 12), remplace trim en déduisant le trimestre correspondant.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères du type "troisième trimestre 2023" ou "3^e trimestre 2023".

See Also

[nextTrim](#), [prevTrim](#)

Examples

```
quelTrim(3, 2023)  
quelTrim(3, 2023, majuscule = TRUE)  
quelTrim(3, 2023, type = "chiffres")  
quelTrim(1, 2023, type = "chiffres")  
quelTrim(999, 2023, type = "chiffres", mois = 8)  
quelTrim(3, 2023, type = "chiffres", exposant = FALSE)
```

s*Met au pluriel selon la valeur numérique*

Description

Retourne la forme singulière ou plurielle selon la valeur absolue du nombre.

Usage

```
s(a, sing = "", plur = "s", seuil = 2)
```

Arguments

<code>a</code>	Valeur numérique.
<code>sing</code>	Forme au singulier. Par défaut : chaîne vide.
<code>plur</code>	Forme au pluriel. Par défaut : "s".
<code>seuil</code>	Seuil à partir duquel la forme plurielle est utilisée (par défaut : 2).

Value

Une chaîne de caractères correspondant à la forme correctement accordée. NA si l'entrée est NA.

See Also

<https://www.dictionnaire-academie.fr/article/QDL057>

Examples

```
serad::s(-7.5)
serad::s(-2)
serad::s(1.4, "chat parle", "chats parlent")
serad::s(-2, "chat parle", "chats parlent")
serad::s(1.97)
serad::s(arrondi_tot(1.97))
serad::s(1.97, seuil = 1.95)
```

set_serad_language	<i>Définit la langue de serad</i>
--------------------	-----------------------------------

Description

Modifie la langue utilisée par {serad} et réinitialise les règles de rédaction associées.

Usage

```
set_serad_language(lang = c("fr", "en"))
```

Arguments

lang	Langue de sortie : "fr" ou "en".
------	----------------------------------

Value

NULL invisiblement.

whichMois	<i>Récupère le numéro du mois à partir d'un texte</i>
-----------	---

Description

Extrait le numéro du mois (1 à 12) à partir d'une chaîne de caractères contenant un mois en format abrégé ou littéraire.

Usage

```
whichMois(mois)
```

Arguments

mois	Chaîne de caractères contenant un mois (ex. "En juil. 98").
------	---

Value

Un entier entre 1 et 12 si un mois est détecté, 0 sinon. NA si l'entrée est NA.

See Also

[quelMois](#)

Examples

```
whichMois("En juil. 98")
```

Index

a, [3](#), [15–17](#), [19](#)
alahausse, [4](#)
arrondi_tot, [4](#), [11–13](#)
audessus, [5](#)

comparaison, [4](#), [5](#), [6](#), [7–10](#)
comparaison_taux, [7](#), [7](#)
contributions, [8](#)

davantage, [9](#)
depasse, [10](#)

format_delta, [10](#)
format_g, [3](#), [11](#), [13](#), [14](#), [26](#)
format_niv, [5](#), [11](#), [12](#)
format_pts, [13](#), [26](#)

g, [3](#), [6](#), [7](#), [12](#), [14](#), [16](#), [18](#), [21](#), [24](#)
g_nom, [20](#)
g_nom_simple, [21](#)
g_nom_taux, [21](#), [22](#), [26](#), [27](#)
g_verbe, [23](#), [26](#)
g_verbe_taux, [23](#), [24](#), [25](#), [26](#), [27](#)
get_serad_language, [20](#)
gETa_nom, [15](#)
gETa_nom_taux, [15](#), [16](#), [16](#), [19](#), [26](#), [27](#)
gETa_verbe, [17](#)
gETa_verbe_taux, [17](#), [18](#), [18](#), [26](#), [27](#)

init_serad, [15–19](#), [21](#), [23–26](#), [26](#)
init_serad_en (init_serad), [26](#)
init_serad_fr (init_serad), [26](#)

nextMois, [27](#), [31](#), [33](#)
nextTrim, [28](#), [28](#), [32](#), [34](#)

plushautniveau, [29](#)
prevMois, [28](#), [30](#), [33](#)
prevTrim, [29](#), [31](#), [31](#), [34](#)

quelMois, [28](#), [31](#), [32](#), [36](#)

quelTrim, [29](#), [32](#), [33](#), [33](#)

s, [35](#)
set_serad_language, [36](#)

whichMois, [36](#)